UNIVERSITY OF WISCONSIN–MADISON
COMPUTER SCIENCES DEPARTMENT

CS 564: DATABASE MANAGEMENT SYSTEMS

Assignment 3: B+ Trees: BTPage
Due: Monday, October 16, 1995, 5 p.m.

Instructor: Raghu Ramakrishnan

1 Introduction

In this assignment, you will implement the page level structure of a B+ tree, in which leaf level pages contain entries of the form ⟨*key, record–id*⟩. The key can be either a string or an integer. The RecordPage class is essentially the same as HFPage, with the addition of three functions. These are explained in the FAQ. The documentation available in the header files is sufficient to understand what operation each function performs.

This assignment involves the implementation of 3 classes.

1. **SortedPage** — It is derived from RecordPage. It augments the `insertRecord` method of RecordPage by storing records on the RecordPage in sorted order by the *key* value.

2. **BTIndexPage** — It is derived from SortedPage, and stores *index entries* (a key and PageId) on the SortedPage, as well as a pointer to the left-most child page.

3. **BTLeafPage** — Also derived from SortedPage. It stores *data entries* (a key and the record–id of a data record) on the SortedPage, as well as left- and right- sibling pointers. (Note that the Minibase implementation of B+ trees that you used in the previous assignment did not set these sibling pointers correctly! You, of course, should do this right.)

You will implement the key values using `void *` pointers. The contents of a key will be interpreted using the `KeyType` variable. The various types of keys are in `bt.h`

The results of this assignment will be used by you for the next assignment. So make sure it works! *You will carry out this assignment in teams of two with the same partner as for the previous assignment.*

2 Available Documentation

You should begin by reading the chapter on Tree Structured Indexes, in particular the sections on the structure of the index and leaf pages in a B+ tree. The header files for the three classes cover everything else that you need in order to implement them.

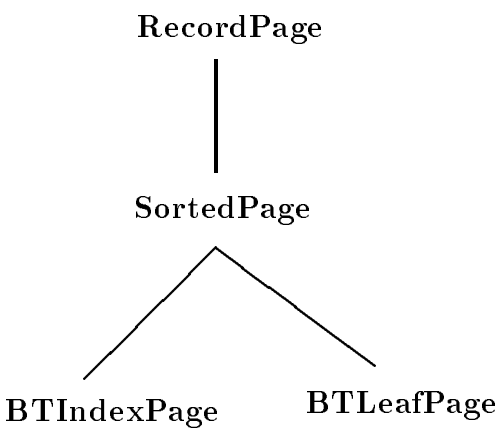

Figure 1: The C++ Classes used for the B+Tree pages

3 Getting Started

Copy all the files from `/p/course/cs564-raghu/assigns95/assign3/` to your own local assignment 3 directory and study them carefully. The files are:

- *Makefile:* A sample Makefile to compile your project. You will have to set up any dependencies by editing this file. You can also design your own Makefile.
- Partial Templates for: *SortedPage, BTIndexPage, BTLeafPage* classes. You must NOT add any private data members to these classes. You have to implement the specification of these classes as part of this assignment.
- Look at the error protocol document at:
  `http://www.cs.wisc.edu/coral/minibase/system/error.html`
  This is the error protocol that will be used. *You* **must** *adhere to it.*

3.1 Overview of the Classes

I will assume that everyone understands the concept of B+ trees, and the basic algorithms, and will concentrate on explaining the design of the C++ classes you will implement.

To understand the classes that you will implement, it is best to begin by looking at the class structure shown in Figure 1. Note again that you must not add any private data members to *SortedPage,*

BTIndexPage or *BTLeafPage*. For further details about the individual methods in these classes, look at the header pages for the class.

3.2 SortedPage

This class is derived from the class RecordPage. Its only function is to maintain records on a RecordPage in a sorted order. Only the slot directory is re-arranged during a sorted insertion. The data records remain in the same positions on the page. You will want to use *RecordPage::compact_slot_directory()* to remove any holes in the slot directory that may have resulted from the deletion of a record by RecordPage::deleteRecord().

3.3 BTIndexPage

This class is derived from SortedPage. It inserts records of the type ⟨*key, pageNo*⟩ on the SortedPage. The records are sorted by the key.

3.4 BTLeafPage

This class is derived from SortedPage. It inserts records of the type ⟨*key, dataRid*⟩ on the SortedPage. *dataRid* is the Rid of the data record. The records are sorted by the key.

4 Compiling Your Code and Running the Tests

Only four tests provided. Feel free to add more tests until you are convinced that your code works. (Remember that you will be using it for the next assignment!)

5 What to Turn In, and When

Electronically, turn in copies of your **output**. Submit hard copies of your .C and .h files (only those which you have modified!) to the TAs. The assignment is due at 5PM on October 16th. The solution will be made public after that time, and solutions turned in after that time will **not** receive any credit. So make sure to turn in whatever you have, even if it is not working fully, at that time.